

REMARKS

As a preliminary matter, nonelected claims 26 and 27 have been canceled, without prejudice.

As another preliminary matter, applicant appreciates the indication that claims 25-28 are directed to allowable subject matter.

Claims 1, 6, 25 and 28 have been amended for cosmetic reasons unrelated to patentability, without narrowing the scope of the claims.

Claims 3 and 14 have been amended to address the Examiner's objections under 35 USC § 112 in paragraphs 3(a) and (b) of the Office Action. The further objection to claims 3, 21 and 25 in paragraph 3(c) is traversed. In the preferred embodiment of the invention, the information contained in a value-requiring instruction is a so-called "sequence offset" of the kind defined in claim 4. However, as explained in the original description at page 34, lines 2 to 7, it is also possible to specify the identity of the register holding the required value using its assigned sequence number directly or relative to some reference point other than the sequence number currently reached. The words "dependent upon said sequence number" encompass these possibilities disclosed in the original specification. Withdrawal of this objection is respectfully requested.

Claims 1-3, 5-7, 15, 16 and 28 stand rejected under § 102 on the basis of Wang et al. '055. Applicant traverses this rejection because Wang does not disclose (or suggest) the sequence number assigning unit of independent claims 1 and 28.

Wang discloses a processor having a look-ahead capability, i.e. the ability to examine instructions beyond the current point of execution in an attempt to find independent instructions for immediate execution. In such a processor, instructions are executed out of order at times. This out-of-order execution leads to complications in dealing with situations in which the execution of the instructions is interrupted or altered, so that the execution must be restarted in the correct order. Such situations include branch predication failure and exceptions such as interrupts. Wang discloses an instruction retirement unit (400 in Figure 4) which retires instructions and updates the state of the machine such that when a restart is required due to an exception or a branch misprediction, the current state up to that point is recoverable without needing to wait for the register file to be rebuilt or reconstructed, to negate the effects of out-of-order executions (Wang column 7, lines 61 to 67).

In Wang, if an instruction is completed out of order, and there are previous instructions that have not been completed, the results of that instruction are temporarily stored in a temporary buffer 403 formed within the register file 102 (column 8, lines 21 to 24). These temporarily-stored results are only transferred from the temporary buffer 403 to the "real" register array 404 when the instruction in question is retrievable, that is, when the instruction itself is completed and there are no unexecuted instructions appearing earlier in the program order (column 8, lines 24 to 32). If the result of an executed instruction is required by a subsequent instruction, that result will therefore be held in the temporary buffer 403 if the executed instruction is not yet retired, and will be in the register array 404 if that

instruction is considered retired. The functional units 104 outside the register file 102 need not know whether the result is in the temporary buffer 403 or in the register array 404; this is an internal matter for the instruction retirement unit 400 (column 8, lines 33 to 36 and column 11, line 58 to column 12, line 7).

The processor in Wang has an instruction buffer which is a FIFO register containing instructions to be executed. As instructions are completed, they are flushed out at the bottom and new instructions are dropped in at the top. The bottom two locations in the instruction buffer are referred to as an instruction window 202 (column 6, lines 20 to 32). The instruction window 202 defines the instructions which may issue during one clock cycle. In the particular implementation example given, a maximum of eight pending instructions can be contained in the instruction window 202. Each instruction in the instruction window 202 is assigned its own "tag" according to the instruction's location in the instruction window 202 (see Table 1 in column 9 and column 9, lines 8 to 19). The tag denotes one of eight locations in the temporary buffer 403. When an instruction in the instruction window 202 is executed and its results are output from a functional unit, the tag follows the result. If the instruction is one which has been executed out of order, it is stored in the temporary buffer 403 in the storage location thereof identified by the tag (column 9, lines 47 to 50 and column 10, lines 6 to 13).

The Examiner appears to be equating the tags in Wang with the sequence numbers of claim 1. The tag assigned to an instruction in Wang identifies a temporary

storage location for the result of that instruction, but in Wang each one of the instructions in the instruction window 202 is assigned a tag, whether the instruction is a value-producing instruction which does need a register to store its result or is some other kind of instruction which has no such need. Many instructions, for example, load instructions and branch instructions, do not produce results and therefore have no need for a register at all. In Wang, therefore, many of the storage locations identified by the tags do not hold actual results. This waste of resources is acceptable in Wang because the storage locations are only temporary storage locations equal in number (8 in the example) to the maximum number of instructions that can be issued in any one clock cycle.

In the present invention, on the other hand, the sequence numbers are assigned directly to the results of value-producing instructions (as opposed to all instructions as in Wang), and sequence numbers are not "wasted" by assigning them to instructions which do not require registers to store their results.

The significance of this difference will now be explained. A program for execution by a processor embodying the present invention includes at least one value-requiring instruction which, when executed, requires the produced value of a previously-issued value-producing instruction. There may be any number of instructions between the value-producing instruction and the value-requiring instruction, so it is extremely wasteful to assign sequence numbers to all the intermediate instructions, as the processor will have to have a different register for each different sequence number assigned. In the present

invention, a new sequence number is assigned when a new value-producing instruction is issued, so there is no waste of resources.

This scheme of assigning the sequence numbers to the produced values also provides advantages in relation to the compiled program which the processor executes. In all processors, a value-requiring instruction must contain information which the processor uses to identify the particular register in the register file which has been used to store the produced value. It is the responsibility of the compiler to produce this information. In the present invention, the information in the value-requiring instruction is dependent on the sequence number of the produced value of the previously-issued value-producing instruction. For this scheme to work efficiently, a new sequence number should only be assigned each time a value will be produced. This only happens when a value-producing instruction is issued. If a new sequence number is issued each time any instruction is issued (as in Wang), many more sequence numbers would have to be assigned, increasing the size of the information that would need to be contained in each value-requiring instruction. In fact, in this respect, Wang is even more remote from the present invention, as the value-requiring instructions do not even contain information dependent on the tags. The tags are merely generated at the time of issuance of the instructions and used internally by the instruction retirement unit 400. The compiler in Wang does not identify registers using the tags.

Thus, for the reasons set out above, Wang does not disclose a sequence number assigning unit which assigns the values produced by the value-producing instructions

respective sequence numbers according to the order of issuance of their respective value-producing instructions, as required by independent claims 1 and 28. Also, since Wang is only concerned with a very simple scheme to buffer the results of the eight instructions in the instruction window, it is perfectly feasible to employ one tag per instruction, without the numbers of tags proliferating. A skilled worker would therefore have no motivation to try to modify the way in which the tags are assigned to assign tags only to instructions which do produce values. Thus, Wang does not render obvious the present claims 1 and 28, either. The rejected dependent claims are allowable for the same reason. Accordingly, withdrawal of this rejection is respectfully requested.

For the foregoing reasons, applicant believes that this case is in condition for allowance, which is respectfully requested. The examiner should call applicant's attorney if an interview would expedite prosecution.

Respectfully submitted,

GREER, BURNS & CRAIN, LTD.

By 

By

Patrick G. Burns
Registration No. 29,367

April 29, 2004

300 South Wacker Drive
Suite 2500
Chicago, Illinois 60606
Telephone: 312.360.0080
Facsimile: 312.360.9315
P:\DOCS\0808\65202\518901.DOC